

Logic Programming Engineering – Assignment 6

Dr.-Ing. Sascha Klüppelholz

Exercise 6.1 [Scalar product]

Define a predicate `scalar/3` which for two given vectors $\vec{v}_1 = (a_1, a_2, \dots, a_n)$ and $\vec{v}_2 = (b_1, b_2, \dots, b_n)$ of integers (i.e., $a_i, b_i \in \mathbb{Z}$, $i = 1, \dots, n$) computes the scalar product. Please remind: The scalar product (aka dot product) of $\vec{v}_1$ and $\vec{v}_2$ is defined as $\vec{v}_1 \cdot \vec{v}_2 = \sum_{i=1}^n a_i b_i$.

Solution:

```
scalar([X], [Y], S) :-  
    S is X*Y, !.  
  
scalar([X|List1], [Y|List2], S) :-  
    scalar(List1, List2, S1),  
    S is S1 + X * Y.
```

Exercise 6.2 [Polynomials]

Polynomials can be represented as lists of pairs of coefficients and exponents. For example the polynomial $4x^5 + 2x^3 - x + 27$ can be represented as the following Prolog list: `[(4,5), (2,3), (-1,1), (27,0)]`.

a) Provide a predicate `polyValue/3` that computes the polynomial for a given value of the variable $x \in \mathbb{R}$.

Solution:

```
polyValue(Value, [(Coefficient, Exponent)], Result) :-  
    Result is Coefficient * (Value^Exponent), !.  
  
polyValue(Value, [(Coefficient, Exponent) | Tail], Result) :-  
    T1 is Coefficient * (Value^Exponent),  
    polyValue(Value, Tail, T2),  
    Result is T1 + T2.
```

b) Now provide an alternative predicate called `horner/3` that does the same as `polyValue/3`, but follows the Horner scheme to compute the polynomial for a given value of the variable $x \in \mathbb{R}$, e.g.,

$$3x^3 + 2x^2 - 5x + 4 = 4 + x(-5 + x(2 + x(3)))$$

For this task you may, e.g., assume that the polynomial is given as a complete and sorted list whose order is increasing (or decreasing) in the exponents of the polynomial, i.e., the polynomial $4x^5 + 2x^3 - x + 27$ would be given as the Prolog list [(27,0), (-1,1), (0,2), (2,3), (0,4), (4,5)].

Solution:

```
horner(_, [(Coefficient, _)], Coefficient):- !.
```

```
horner(Value, [(Coefficient, _) | Tail], Result):-
    horner(Value, Tail, T2),
    Result is (T2*Value)+Coefficient.
```

- c) Create a random generator for polynomials with a given maximum degree and maximum coefficient size. For this, provide a predicate randomPoly/3. The format of the list should be compatible with the parts a) and b) of this exercise.

Solution:

```
randomPoly([(C,0)], 0, S):-
    C is random(S), !.
```

```
randomPoly(L,N,S):-
    NPrime is N-1,
    randomPoly(LPrime, NPrime, S),
    C is random(S),
    append(LPrime, [(C,N)], L).
```

- d) Provide a predicate polyTester using randomPoly/3 from part c) to create random polynomials and compute the polynomial for random values of $x \in \mathbb{N}$ with the help of polyValue/3 and horner/3. Use the new predicate for comparing the two methods with respect to their efficiency. Hint: you may for example use the predicate time/1 or the SWI-Prolog profiler (cf. profile/1).

Solution:

```
polyTester(MaxDegree, MaxCoeff, MaxValue) :-
    randomPoly(P, MaxDegree, MaxCoeff),
    write('Poly: '), writeln(P),
    X is random(MaxValue)+1,
    write('Random value x: '), writeln(X),
    time(horner(X,P, Result)),
    time(polyValue(X,P, Result)),
    write('Polyvalue for x= '), write(X), write(' : '),
    writeln(Result).
```

- e) Write a predicate polysum/3 for adding two polynomials. Your solution should work independent of the ordering of pairs inside the two given lists.

Solution:

```
polysum([], List, List) :- !.
```

```
polysum([(X1,Y) | Tail], List, PolySum) :-
    memberchk((X2,Y), List),
```

```

    select ((X2,Y), List , ListPrime), !,
    X is X1+X2,
    polysum(Tail , ListPrime , PolySumPrime),
    append([(X,Y)], PolySumPrime , PolySum).

polysum([(X1,Y) | Tail], List , PolySum) :-
    polysum(Tail , List , PolySumPrime),
    append([(X1,Y)], PolySumPrime , PolySum).

```